Application Brief

Using Serial Communications Within TPLD

Owen Westfall

Why Use Serial Communications

The serial communication in TI's Programmable Logic Device (TPLD) enables the use of the USER register space. Using the USER register space allows for small adjustments of select fields, like counter blocks control data field, or can be used similar to I/O without having to go through device pins. The TPLD is currently only capable of being a The USER registers can be found under Detailed Description -> Device functional modes -> Programming -> “Device name” Registers -> “device name”_USER_REGISTER. For this application brief, the TPLD1202 is the device being used and any references to addresses are based on that devices USER register table.

Note

The USER register space is not the same as the configuration space. The USER register space does not allow for changing connections, or modifying the original design beyond select fields.

What is Available in the Register Space

Table 1 is an example of a USER register space. The table does not have all the options available in all devices, but is a large set of what can be available.

Table 1. Example USER Registers

Acronym	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DEVICE_ID0	DEVICE_ID_MSB							
DEVICE_ID1	DEVICE_ID_LSB							
DEVICE_ID2	DEVICE_ID_RSVD							
DEVICE_ID3	DEVICE_ID_REV							
DEVICE_ID4	DEVICE_ID4							
DEVICE_ID5	DEVICE_ID5							
DEVICE_ID6	DEVICE_ID6							
DEVICE_ID7	DEVICE_ID7							
CNT0_COUNT	CNT0_COUNT							
CNT1_COUNT	CNT1_COUNT							
CNT2_COUNT	CNT2_COUNT							
CNT3_COUNT	CNT3_COUNT							
CNT4_COUNT_LSB	CNT4_COUNT_LSB							
CNT4_COUNT_MSB	CNT4_COUNT_MSB							
CNT5_COUNT_LSB	CNT5_COUNT_LSB							
CNT5_COUNT_MSB	CNT5_COUNT_MSB							
CNT6_COUNT	CNT6_COUNT							
CNT7_COUNT	CNT7_COUNT							
CNT8_COUNT	CNT8_COUNT							
CNT9_COUNT	CNT9_COUNT							
CNT0_DATA	CNT0_DATA							
CNT1_DATA	CNT1_DATA							
CNT2_DATA	CNT2_DATA							
CNT3_DATA	CNT3_DATA							
CNT4_DATA_LSB	CNT4_DATA_LSB							
CNT4_DATA_MSB	CNT4_DATA_MSB							
CNT5_DATA_LSB	CNT5_DATA_LSB							

Table 1. Example USER Registers (continued)

Acronym	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CNT5_DATA_MSB	CNT5_DATA_MSB							
CNT6_DATA	CNT6_DATA							
CNT7_DATA	CNT7_DATA							
CNT8_DATA	CNT8_DATA							
CNT9_DATA	CNT9_DATA							
WATCHDOG_TIMEOUT_DATA	WATCHDOG_TIMEOUT_DATA							
WATCHDOG_OUTPUT_DATA	WATCHDOG_OUTPUT_DATA							
WATCHDOG_STATUS	WATCHDOG_STATUS							
PGEN_DATA_LSB	PGEN_DATA_LSB							
PGEN_DATA_MSB	PGEN_DATA_MSB							
STATE_MACHINE	RESERVED					CURRENT_STATE		
STATE0_OUT	STATE0_OUT							
STATE1_OUT	STATE1_OUT							
STATE2_OUT	STATE2_OUT							
STATE3_OUT	STATE3_OUT							
STATE4_OUT	STATE4_OUT							
STATE5_OUT	STATE5_OUT							
STATE6_OUT	STATE6_OUT							
STATE7_OUT	STATE7_OUT							
VREF_ACMP0	VREF_ACMP0							
VREF_ACMP1	VREF_ACMP1							
VREF_ACMP2	VREF_ACMP2							
VREF_ACMP3	VREF_ACMP3							
VREF_McACMP0_0	VREF_McACMP0_0							
VREF_McACMP0_1	VREF_McACMP0_1							
VREF_McACMP1_0	VREF_McACMP1_0							
VREF_McACMP1_1	VREF_McACMP1_1							
VREF_McACMP2_0	VREF_McACMP2_0							
VREF_McACMP2_1	VREF_McACMP2_1							
VREF_McACMP3_0	VREF_McACMP3_0							
VREF_McACMP3_1	VREF_McACMP3_1							
VIRTUAL_INPUT	VIRTUAL_IN							
VIRTUAL_OUTPUT	VIRTUAL_OUT							
SER_COMM_CFG	RESERVED							ADDR_AUTOINC
CRC_STATUS	CRC_ERR_CNT			RESERVED				CRC_ERR_FLAG
SER_COMM_WR_MASK	SER_COMM_WR_MASK							

The DEVICE_ID registers are used to create an identity for each design. The first 2 DEVICE_ID registers are set by the device depending on the device selected for example the TPLD1202 has 0x12 in DEVICE_ID0 and 0x02 in DEVICE_ID1 where as the TPLD2001 has 0x20 in DEVICE_ID0, and 0x01 in DEVICE_ID1. DEVICE_ID 4 - 7 are loaded at startup with the Program ID written in the system settings within InterConnect Studio.

Starting from address 0 to address 7 are identifiers of the device. This includes the device id which based on the TI product number, for example. The TPLD1202 has register at address 0x00 = 0x12, and address 0x01 = 0x02 where as the TPLD2001 has 0x20, and 0x01 respectively. After that in registers DEVICE_ID 4-7 are loaded at startup with the Program ID. What this means is reading from Addresses 0-7 can have an entirely unique value to each design created by the user. Reading those registers not only provide which device is being used, but also which design is currently being loaded.

The CNTx_COUNT registers are read only registers that reflect whatever the current count of a counter block is. This can be used to give a rough estimate of how long a counter has left before reaching zero. It’s important to note that this read is asynchronous from the actual counter, so depending on when the value is read the counter can iterate before the value is return via the communication protocol.

The CNTx_DATA registers are read/write registers that store the value of control data written in InterConnect Studio. These values can be udated on the fly allowing the user to adjust PWM outputs, Increase/decrease delays, and adjust blocks like the frequency detectors on the fly.

The WATCHDOG portion of the register space allows the user to read the status, adjust the timeout period and output pulse length of the watchdog timer. The WATCHDOG_STATUS records how many times the watchdog has been triggered since startup or since the last read. This register is reset upon being read. Increasing the value in WATCHDOG_TIMEOUT_DATA increases the amount of time a signal can be low before an output pulse is triggered. Increasing the value of WATCHDOG_OUTPUT_DATA increases the length of the output pulse when the watchdog is triggered.

The STATE_MACHINE portion of the register space allows the user to control the current state and adjust each states behavior. SM_CURRENT_STATE contains the current state in binary format in bits 2-0. This is R/W so not only can the current state be read, but the serial controller can be used to force the state machine into certain states. This section also contains the state outputs listed as SM_S#_OUT_CFG. This allows the controller to adjust the outputs of any given state from this section.

the VREF portion of the register space allows the user to adjust the value going into the IN- of both multi-channel analog comparators and independent analog comparators. The value written here is not the value of the VREF, but rather what value the VREF is referencing. In the TPLD2001 the VREF steps by 32mV so increasing the value from 0x00 to 0x01 is changing that reference from 32mV to 64mV.

The VIRTUAL_INPUT register is used to store the values going into the TPLD device. This register is used to act as an optional inputs into the device. This can be used in place of pins to expand the output count of the controller.

The VIRTUAL_OUTPUT register is used to store the values going out of the TPLD device at any given time. Similar to the earlier CNT_COUNT registers mentioned above a read from this register is asynchronous, so after making the request and before the data is returned one of these values can possibly change.

The CRC_STATUS register is used to check if the device has started up correctly. The CRC_ERR_CNT section stores how many times the CRC process iterated before successful startup. The TPLD device is designed to run the CRC_ERR_CNT value up to 8 before full power on. If this value is at 8 and the CRC check fails the CRC_ERR_FLAG is flipped to 1. This register can be used to check if the design in the TPLD was successfully loaded, and if these values are unacceptable power cycle the device. If the device fails start-up again begin error analysis of your design attached to the TPLD.

The SER_COM_WR_MASK register is used to apply a mask over any future reads or writes. This can be used to read just the bottom half, or top half, of a register. The values not within the mask are all be read as 0 by the TPLD or returned as 0 to the controller.

How to Setup I2C or SPI Communication in a System.

Each TPLD has a specific set of pins that are tied to the serial communication which are initialized as the peripheral is added to the design. An example of a I2C setup can be seen in Figure 1, and an example SPI setup in Figure 2.

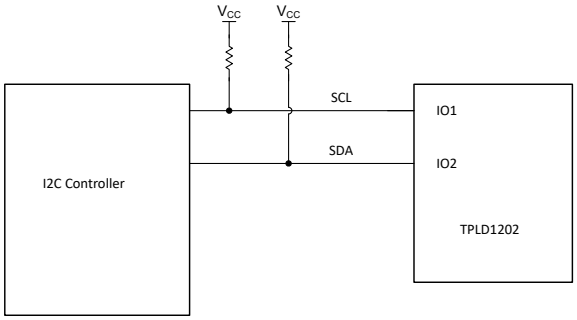

Figure 1. Example I2C Setup with TPLD1202

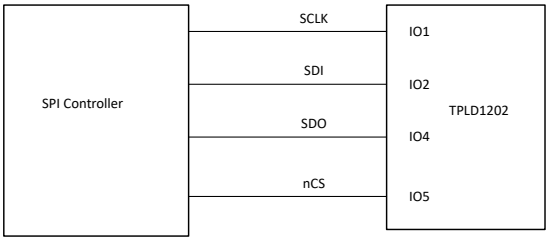

Figure 2. Example SPI Setup with TPLD1202

How to Setup I2C or SPI in InterConnect Studio

InterConnect Studio (ICS) is a software tool used to design, simulate, and configure the TPLD family of devices.

Figure 3 shows the initial setup of the I2C peripheral in InterConnect Studio. I2C can be added to the design by clicking the plus button indicated by the red arrow. The peripheral address is a binary value that can be used to statically set the address of this TPLD design. Below the peripheral address is a setting to enable an external pin-based address. Many I2C peripherals come with predefined addresses, but the TPLD does not, instead using these two settings allows many TPLDs to sit on an I2C bus without interfering with each other. Enabling the pin-based address allows for the peripheral address to be overwritten by the logic value present at the pin when the communication starts. The last setting unique to I2C is the Global Reset Listening which enables the device to be reset when a reset command is sent from the controller.

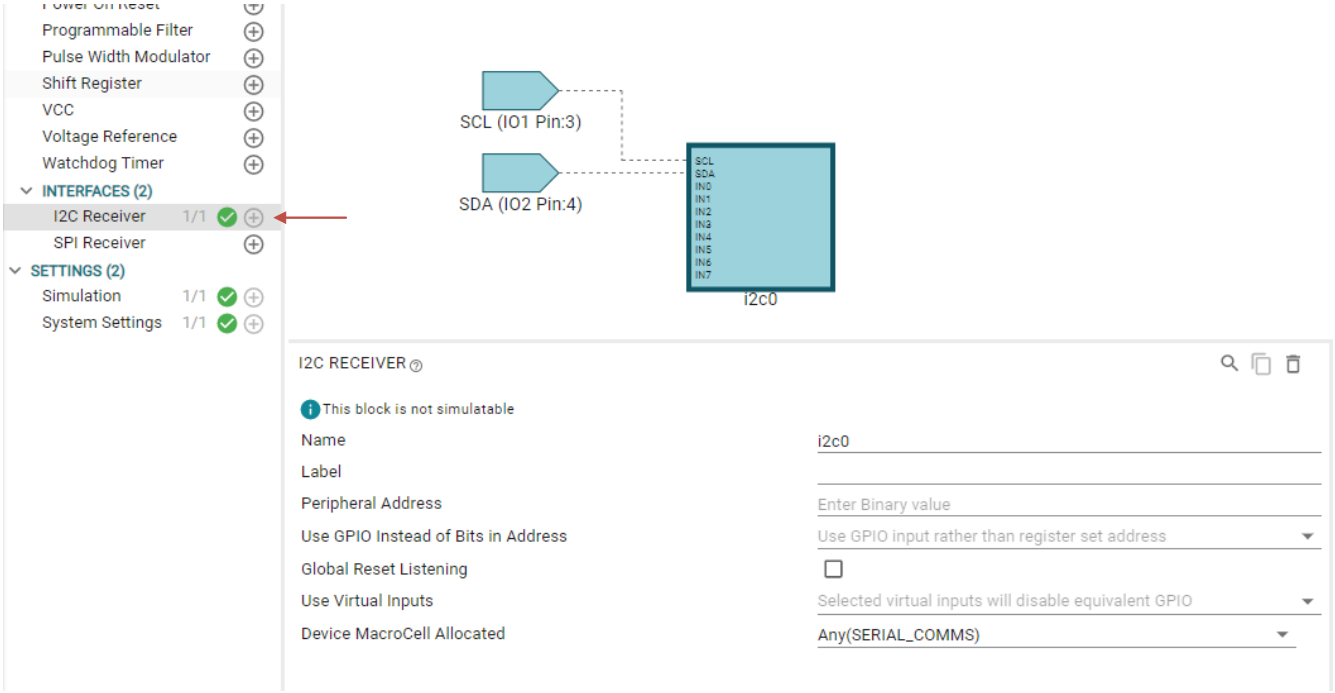

Figure 3. Default I2C in ICS

SPI has fewer options as SPI is not an addressable protocol as shown in Figure 4. The shared option of Virtual Inputs is used to allow the device to view the VIRTUAL_INPUT register. In some TPLD family devices the virtual inputs take the place of some pins as input to the connection matrix, and checking the data sheet clarifies which inputs are unable to be used together. For example in the TPLD1202 VIR_IN0 is shared with IO1 as shown in Table 2, meaning that IO1 cannot be used as an input pin into the design if VIR_IN0 is selected.

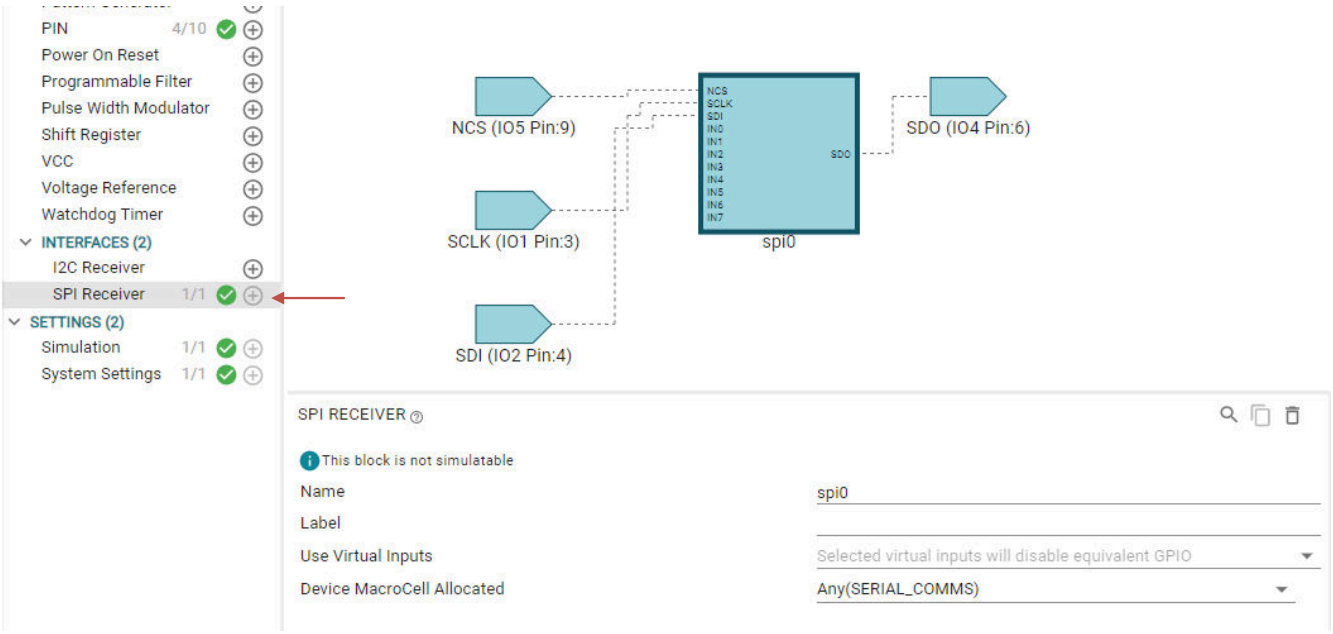

Figure 4. Default SPI in ICS

Table 2. TPLD1202 Shared Inputs

Virtual input	VIR_IN0	VIR_IN1	VIR_IN2	VIR_IN3	VIR_IN4	VIR_IN5	VIR_IN6	VIR_IN7
Digital Input Pin	IO1	IO2	IO3	IO4	IO5	IO6	IO7	IO9

Building a Design Around I2C

A simple implementation of I2C can be using the TPLD1202 as an I/O expander as shown in Figure 5. Three pins are being used as digital inputs and being fed into the input side of the block. These pins are going to feed values into the VIRTUAL_OUTPUT register to be read during operation. Four pins are being used as digital outputs of the device, and are connected to VIR_IN0, VIR_IN1, VIR_IN2, VIR_IN3. These pins reflect the value present in the VIRTUAL_INPUT register, for example, pin4 (IO6) reflects bit 0 so on and so forth.

There is a PWM being run at 50% duty cycle by default. The KEEP input is held high so the duty cycle never changes unless the value of CNT_DATA is overwritten.

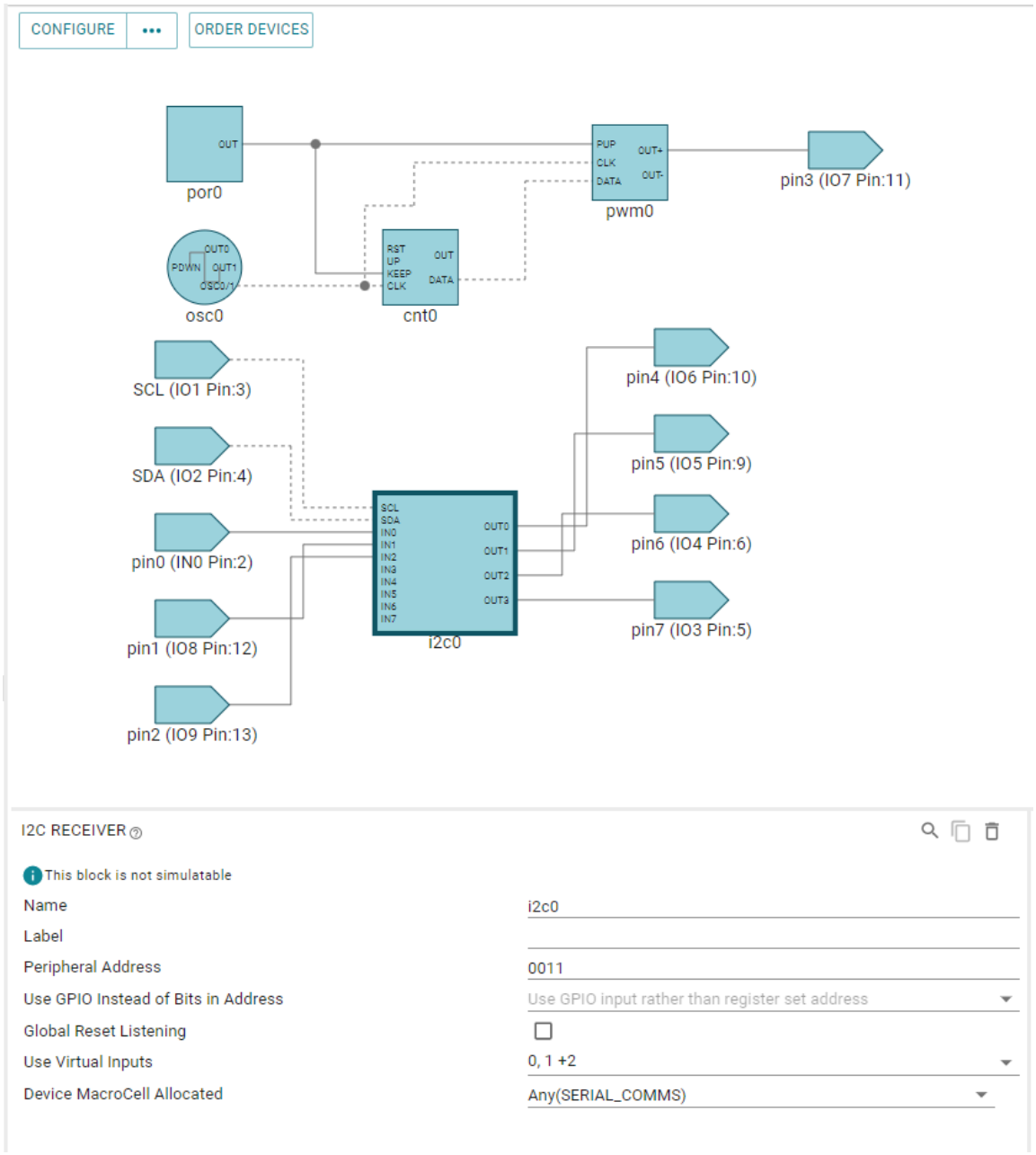

Figure 5. IO Expander With Pulse Width Modulation

The setup for operation is shown in Figure 6 with the logic analyzer connections excluded. The TPLD-PROGRAM is used to update the design without requiring a burn of the device. Internal pull-ups are used within the USB2ANY during messaging.

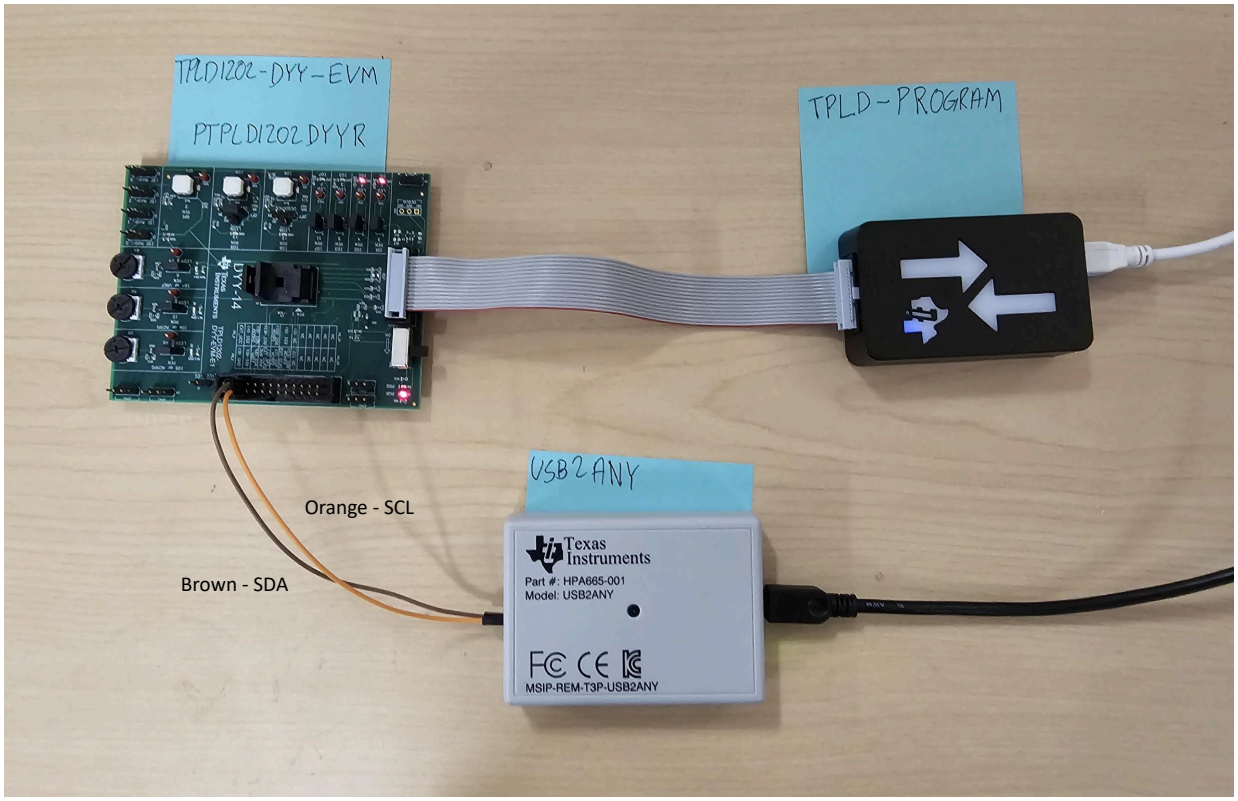

Figure 6. TPLD1202 to a USB2ANY

To write to pins 4 to 7 we write the values to the internal address of VIRTUAL_INPUT (address: 0xE0). The USB2ANY explorer setup can be viewed in Figure 7, and an example of the operation can be viewed in Figure 8. The address entered into the Slave Address box appears different from the design because in the TPLD1202 the I2C receiver only uses 4 bits while the USB2ANY uses a 7 bit address. The address value entered has to be shifted to the left 3 bits from the value entered into ICS for the TPLD to be recognized. The only changes between commands was changing the value in the write data section before selecting Write.

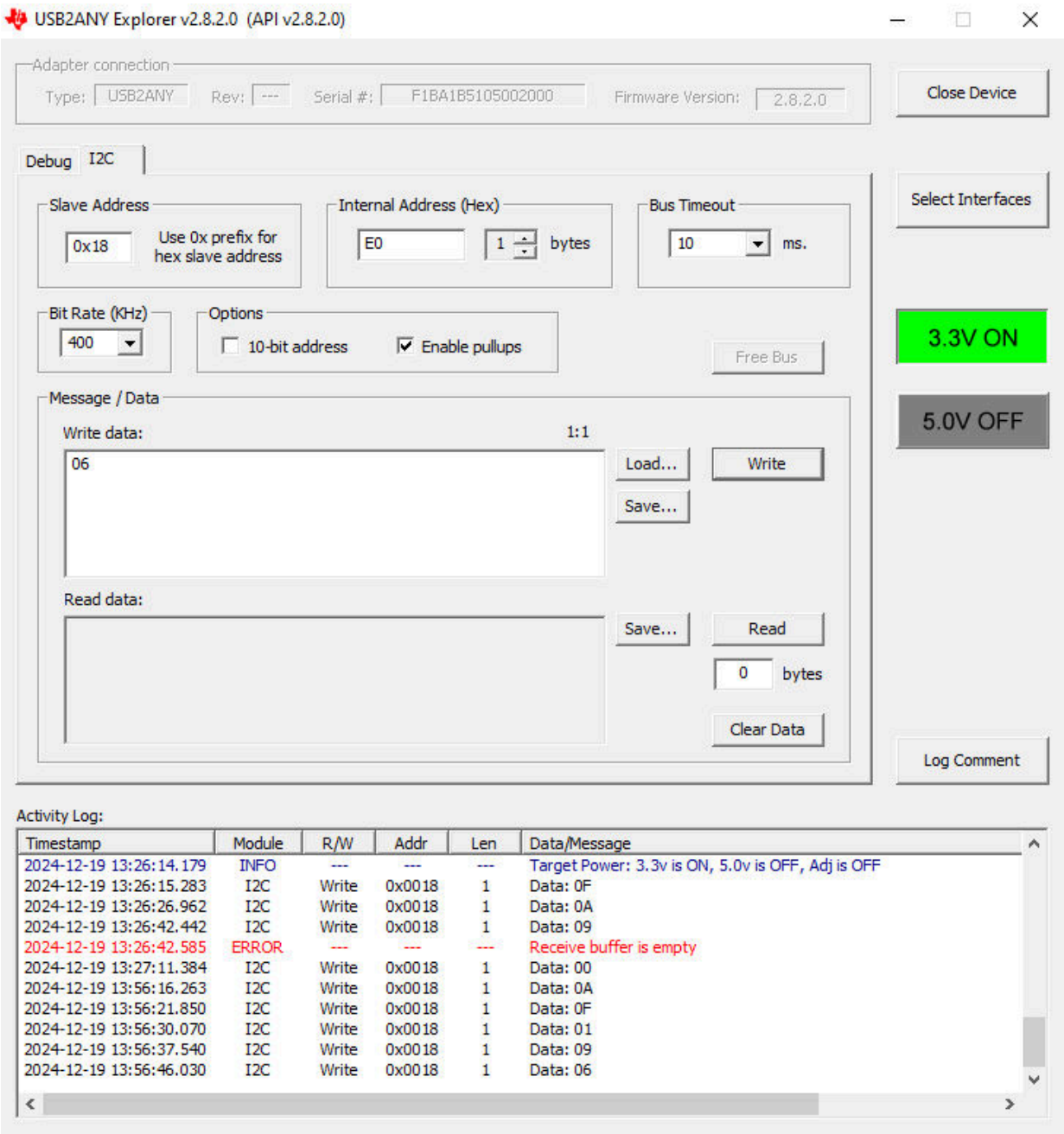

Figure 7. USB2ANY Explorer Write to VIRTUAL_INPUT

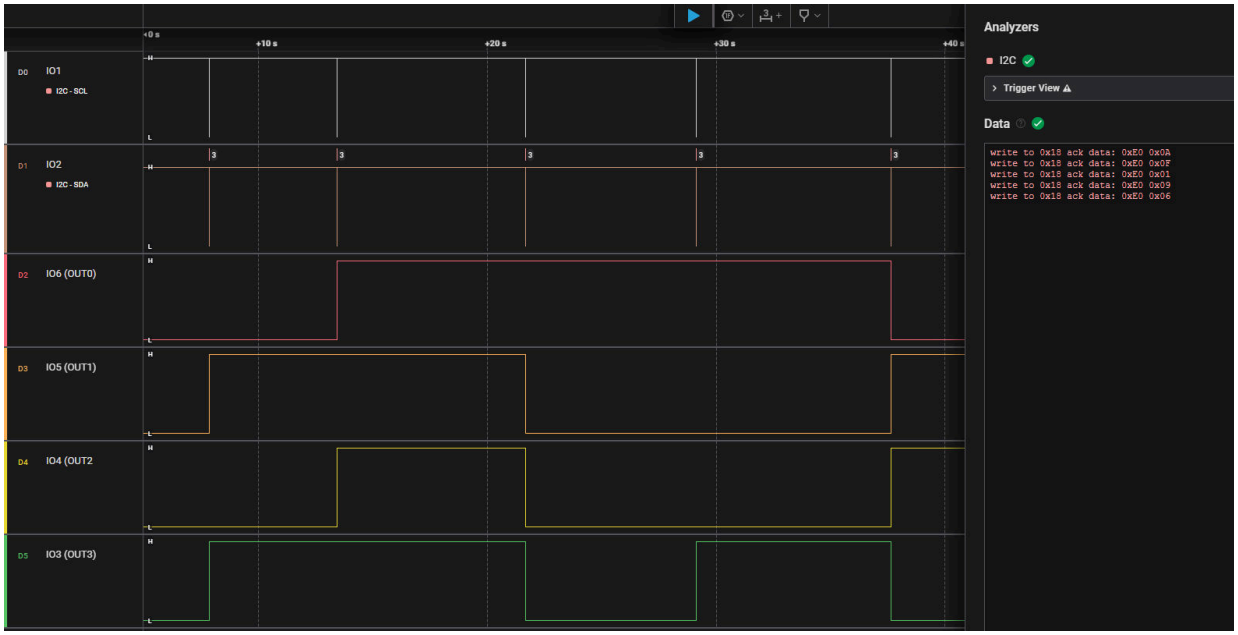

Figure 8. Waveform of Writing to Pins

To read from pins 0 to 2 we first do an empty write to the register VIRTUAL_OUTPUT (address: 0xE1) then execute a read command to the device. The USB2ANY explorer setup can be viewed in Figure 9, and an example of the operation can be viewed in Figure 10

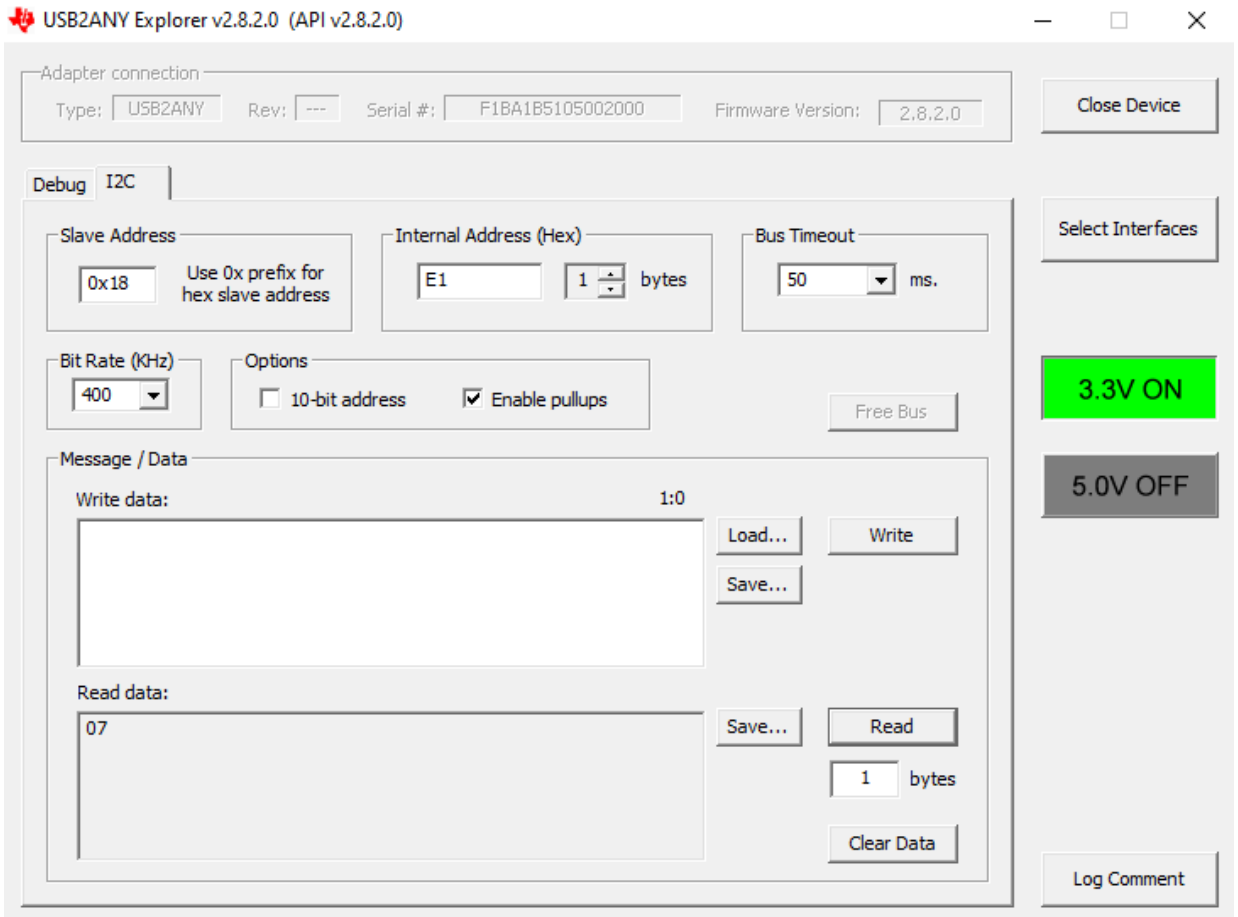

Figure 9. USB2ANY Explorer Read From VIRTUAL_OUTPUT

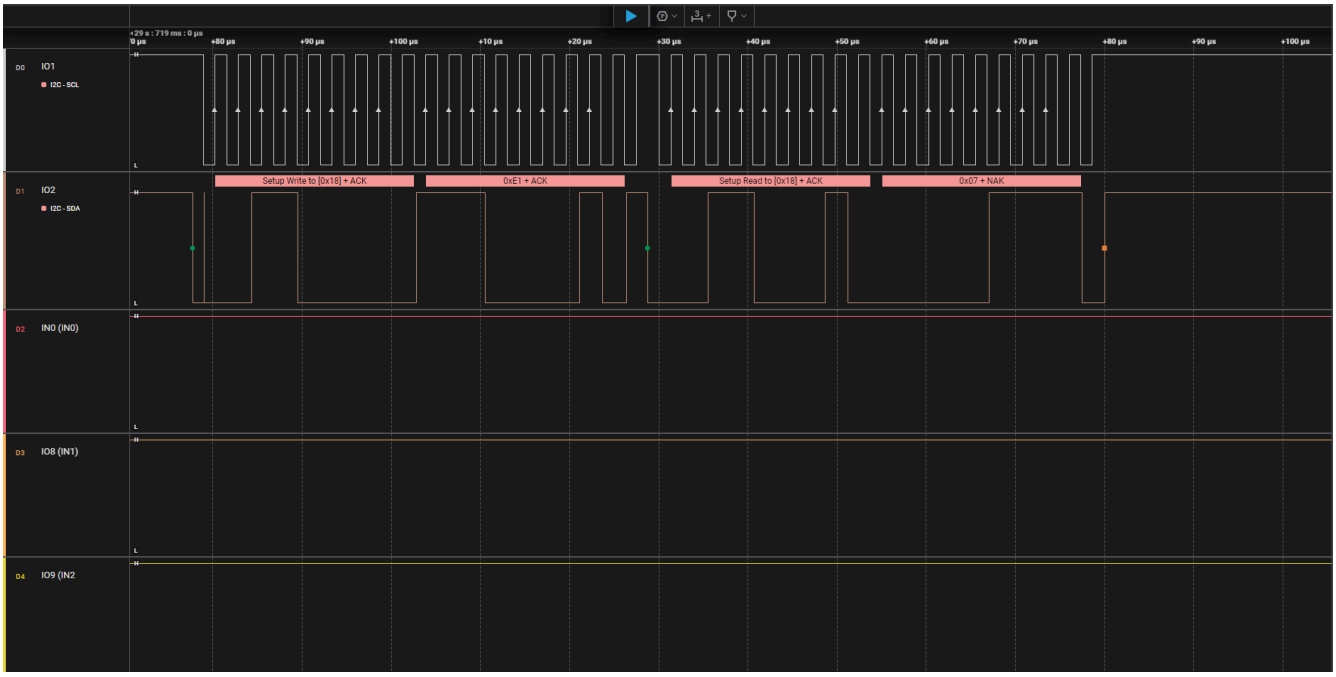

Figure 10. Waveform of Reading From Pins

Lastly to adjust the PWM write a new value to the register CNT6_DATA (address: 0x26). The USB2ANY explorer setup can be viewed in Figure 11, and an example of the operation can be viewed in Figure 12

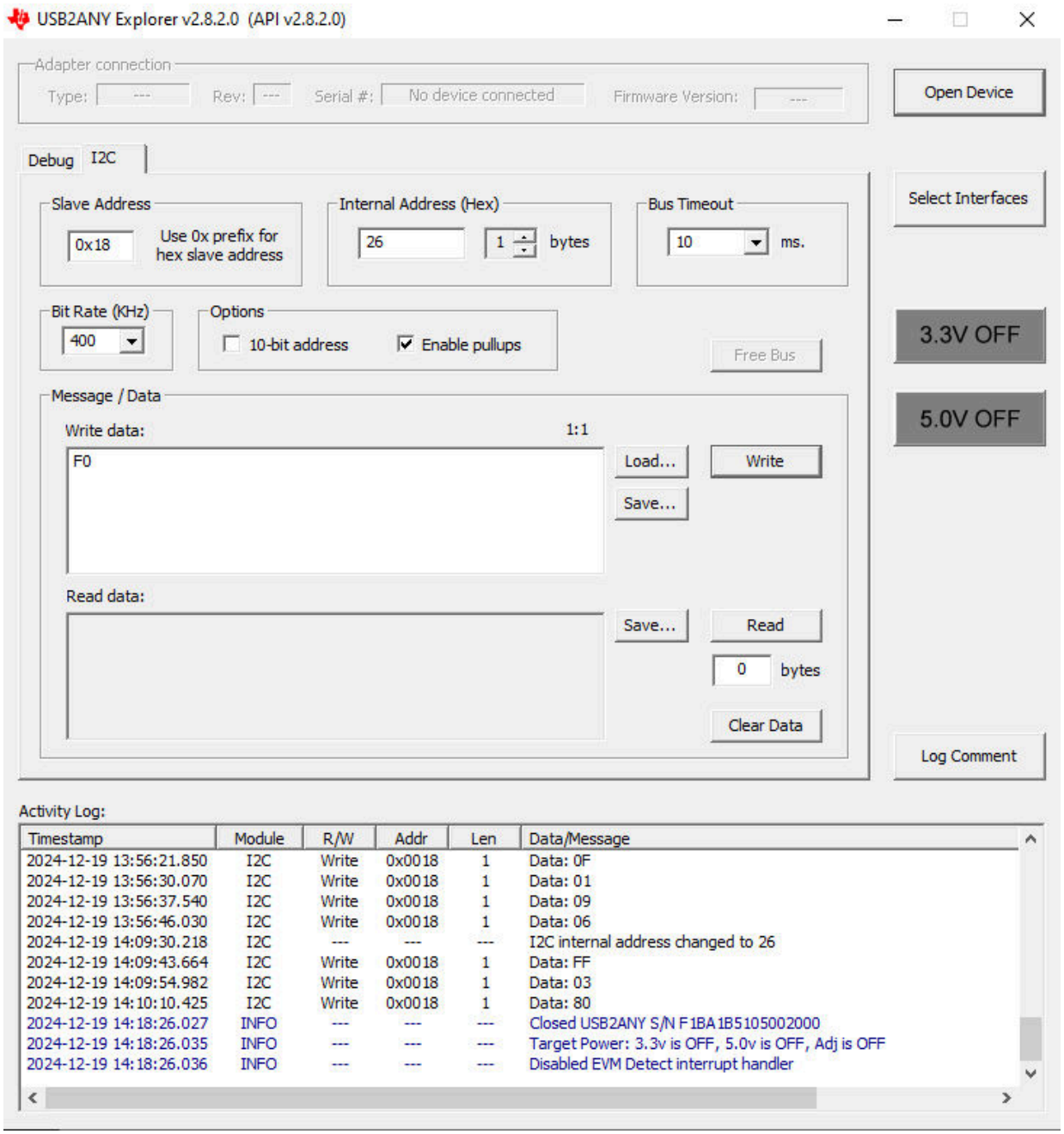

Figure 11. USB2ANY Explorer Write to CNT6_DATA

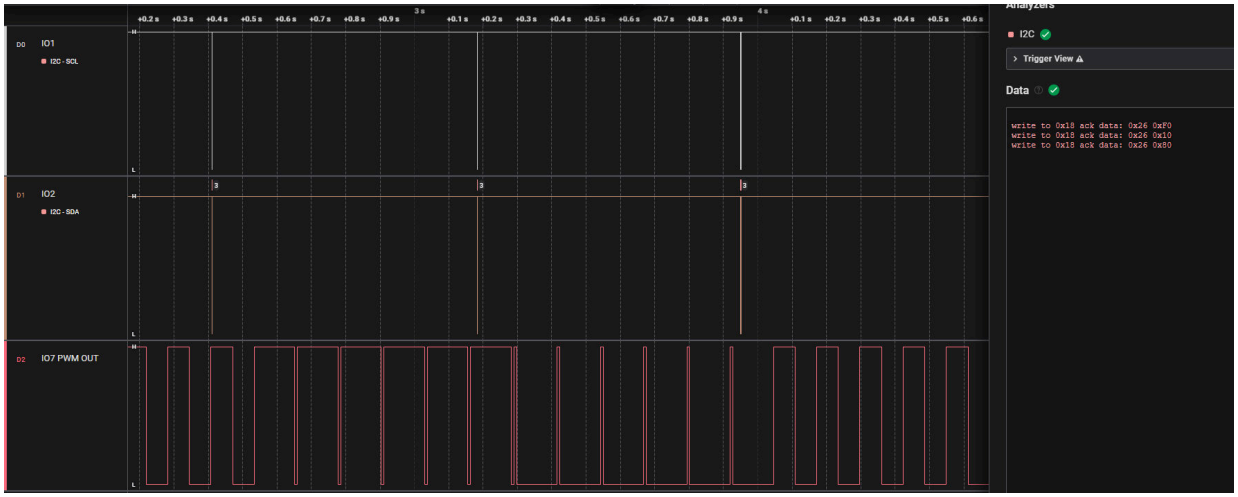

Figure 12. Waveform of Adjusting PWM

Ordering Information

Hardware used in support of this document can be found in Table 3

Table 3. Ordering Information

Device	EVM
All TPLD	TPLD-PROGRAM
TPLD1202	TPLD1202-DYY-EVM TPLD1202-RWB-EVM
N/A	USB2ANY

Trademarks

All trademarks are the property of their respective owners.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265